WHAT IS CLAIMED IS:

1. An information model representing a software architecture, comprising:

a plurality of language independent format objects, each language independent format object representing a discrete component, wherein the components are structurally related into an information model according to a software architecture.

2. The information model of claim 1, wherein a language independent format object comprises:

one or more XML files conforming to a document type definition describing a particular component; and

one or more documentation objects containing information pertaining to the particular component.

3. The information model of claim 1, further comprising:

one or more derivative language independent format objects, wherein a derivative language independent format object represents two or more combined language independent format objects; and

a hierarchical structure relating the plurality of language independent format objects and the derivative language independent format objects into a software architecture.

4. The information model of claim 1, further comprising:

one or more derivative view objects, wherein a derivative view object contains a structural relationship between two or more language independent format objects such that the one or more derivative view objects, in combination with the language independent format objects reflect a software architecture.

5. The information model of claim 1, wherein the software architecture is derived from a body of source code.

6. A system for creating an information model representing an inherent software architecture derived from a body of source code, comprising:

an information model generator having a parser and a composer, the parser configured to extract program fragments from a body of source code and create a plurality of language dependent format objects, the composer configured to convert language dependent format objects into language independent format objects, wherein each language independent format object represents a discrete component in an information model.

7. The system of claim 6 wherein the composer is further configured to create a derivative language independent format object, the derivative language independent format object comprising two or more language independent format objects and representing a discrete component in the information model.

8. The system of claim 6 wherein the composer is further configured to analyze a file system structure of the body of source code and create a derivative view object, the derivative view object relating two or more language independent format objects into a discrete component in the information model based on the file system structure of the body of source code.

9. The system of claim 6 wherein the composer is further configured to analyze a configuration file and create a derivative view object, the derivative view object relating two or more language independent format objects into a discrete component based on the configuration file.

10. A method for creating an information model representing an inherent software architecture derived from a body of source code, comprising:

extracting program fragments from a body of source code;

converting the program fragments to a language independent format; and

creating a plurality of language independent format objects, wherein each language independent format object contains related program fragments from the body of source and represents a discrete component in an information model.

11. The method of claim 10, wherein the related program fragments are related by data dependencies, functional dependencies, and control flow indicators.

12. The method of claim 11, wherein the data dependencies comprise variables passed through function calls and shared global variables.

13. The method of claim 11, wherein the functional dependencies comprise incoming function calls and outgoing function calls.

14. The method of claim 10, further comprising the step of:

creating one or more derivative language independent format objects, wherein a derivative language independent format object comprises two or more language independent format objects.

15. The method of claim 10, further comprising the step of:

creating one or more derivative view objects, wherein a derivative view object relates two or more language independent format objects into a discrete component in the information model.

16. The method of claim 15, wherein the two or more language independent format objects are related based on the file system structure of the body of source code.

17. The method of claim 15, wherein the two or more language independent format objects are related based on a configuration file.

18. A method for creating an information model representing an inherent software architecture derived from a body of source code, comprising:

parsing a body of source code to extract data dependencies, functional dependencies, and control flow indicators;

creating a plurality of language dependent format objects, each language dependent format object comprising an abstract syntax tree representing related program fragments within the body of source code; and

converting each language dependent format object into a language independent format object, wherein each language independent format object represents a discrete component in an information model.

19. The method of claim 18, wherein the related program fragments are related by data dependencies, functional dependencies, and control flow indicators.

20. The method of claim 19, wherein the data dependencies comprise variables passed through function calls and shared global variables.

21. The method of claim 19, wherein the functional dependencies comprise incoming function calls and outgoing function calls.

22. The method of claim 18, further comprising the step of:

creating one or more derivative language independent format objects, wherein a derivative language independent format object comprises two or more language independent format objects.

23. The method of claim 18, further comprising the step of:

creating one or more derivative view objects, wherein a derivative view object relates two or more language independent format objects into a discrete component in the information model.

24. The method of claim 23, wherein the two or more language independent format objects are related based on the file system structure of the body of source code.

25. The method of claim 23, wherein the two or more language independent format objects are related based on a configuration file.

26. A method for creating an information model representing a software architecture, comprising:

creating a plurality of language independent format objects, wherein each language independent format object represents a discrete component in an information model.

27. The method of claim 26, further comprising:

arranging the plurality of language independent format objects into a hierarchical structure representing a software architecture.

28. The method of claim 27, wherein the plurality of language independent format objects are arranged according to data dependencies, functional dependencies, and control flow indicators.

29. The method of claim 28, wherein the data dependencies comprise variables passed through function calls and shared global variables.

30. The method of claim 28, wherein the functional dependencies comprise incoming function calls and outgoing function calls.

31. The method of claim 26, further comprising the step of:

creating one or more derivative language independent format objects, wherein a derivative language independent format object comprises two or more language independent format objects.

32. The method of claim 26, further comprising the step of:

creating one or more derivative view objects, wherein a derivative view object relates two or more language independent format objects into a discrete component in the information model.

33. A system for manipulating an information model representing a software architecture, comprising:

an information model viewer configured to provide a visual presentation of the information model representing the software architecture; and

a system architect configured to modify the software architecture.

34. The system of claim 33, wherein the information model viewer further comprises:

a data dependency viewer configured to present the data dependencies between components of the information model;

a functional dependency viewer configured to present the functional dependencies between components of the information model; and

a calling tree viewer configured to present the control flow between program fragments contained within a component of the information model.

35. The system of claim 34, further comprising:

a search results viewer configured to present the results of searches conducted within one or more information models or within one or more components of one or more information models.

36. The system of claim 33, further comprising:

a language specific viewer configured to provide a visual presentation of the information model representing the software architecture according to one or more programming language paradigms.

37. The system of claim 33, wherein the system architect further comprises:

an architect designer configured to reorganize the hierarchical component structure of the information model.

38. The system of claim 37, wherein the architect designer is further configured to merge two or more components together into a single component.

39. The system of claim 38, wherein the architect designer is further configured to fragment a merged component into two or more components.

40. The system of claim 37, wherein the system architect further comprises:

an architect enhancer configured to add new components to the information model.

41. The system of claim 40, wherein the system architect further comprises:

an architect creator configured to create new components and relate the newly created components into a new information model having no underlying body of source code.

42. The system of claim 41, wherein the system architect further comprises:

an architect optimizer configured to extract functionally related components of an information model and create a new information model having a reduced set of components serving a desired function.

43. The system of claim 33, further comprising:

an information model editor having a text interface and a file interface, the text interface configured to allow editing of documentation associated with the information model and the file interface configured to receive new or modified documentation files associated with the information model.

44. The system of claim 33, further comprising:

an information model builder having a text interface and a file interface, the text interface configured to allow editing of source code files included with the body of source code and the file interface configured to receive new or modified source code files for inclusion with the body of source code.

45. The system of claim 44, wherein the file interface is further configured to provide source code files from the body of source code.

46. The system of claim 33, further comprising:

an information model search engine configured to accept a query, search the information model, and provide search results.

47. The system of claim 33, further comprising:

an information model document generator configured to compile a plurality of documentation objects into an information model document.

48. The system of claim 33, further comprising:

an information model difference generator configured to compare at least two information models and determine the differences between the at least two information models.

49. A method for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, comprising:

merging two or more components into a compound component.

50. The method of claim 49, further comprising:

modifying the hierarchical structure between two or more components;

51. A method for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, comprising:

fragmenting a compound component into its constituent components.

52. The method of claim 51, further comprising:

modifying the hierarchical structure between two or more components;

53. A method for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, comprising:

modifying the hierarchical structure between two or more components in an information model;

merging two or more components into a compound component; and

fragmenting a compound component into its constituent components.

54. The method of claim 53, wherein the modifying step further comprises:

selecting a first component; and

promoting the first component in relation to a second component.

55. The method of claim 53, wherein the modifying step further comprises:

selecting a first component; and

demoting the first component in relation to a second component.

56. The method of claim 53, wherein the merging step further comprises merging a component and a first compound component into a second compound component.

57. The method of claim 53, wherein the merging step further comprises merging a first compound component and a second compound component into a third compound component.

58. The method of claim 53, wherein the fragmenting step further comprises fragmenting a compound component into two or more components.

59. The method of claim 53, wherein the fragmenting step further comprises fragmenting a compound component into a component and a compound component.

60. The method of claim 53, wherein the fragmenting step further comprises fragmenting a compound component into a first compound component and a second compound component.

61. The method of claim 53, further comprising editing documentation associated with the information model through a documentation text editor.

62. The method of claim 53, further comprising uploading new or modified documentation files through a file interface.

63. The method of claim 53, further comprising compiling documentation associated with the information model into an information model document.

64. The method of claim 53, further comprising:

creating a new component in the information model; and

organizing the new component within the hierarchical structure of the information model.

65. The method of claim 53, further comprising searching the information model in response to a query and providing search results.

66. The method of claim 53, further comprising:

comparing a first information model to a second information model; and

generating a difference set containing the differences between the first information model and the second information model identified by the comparison.

67. The method of claim 66, wherein the difference set comprises a new information model.

68. A method for manipulating an information model derived from a body of source code, comprising:

establishing a connection with a server computer;

requesting an information model from the server, wherein the information model is derived from a particular body of source code; and

receiving a visual presentation of the requested information model comprising a plurality of hierarchically arranged components and a plurality of documentation files.

69. The method of claim 68, wherein the plurality of components comprises:

a plurality of single components;

a plurality of compound components, wherein a compound component comprises two or more single components; and

a plurality of data dependencies, functional dependencies, and control indicators, wherein data dependencies, functional dependencies, and control indicators relate the single components and compound components.

70. The method of claim 69, wherein a compound component comprises one or more single components and one or more compound components.

71. The method of claim 69, wherein a compound component comprises two or more compound components.

72. The method of claim 68, further comprising:

viewing a compound component and a sub-component of the compound component;

selecting the sub-component; and

viewing the data dependencies, functional dependencies, and control indicators of the sub-component.

73. The method of claim 68, further comprising:

viewing a documentation file; and

editing the documentation file.

74. The method of claim 68, further comprising:

rearranging the hierarchical structure of the components.

75. The method of claim 68, further comprising:

submitting a search request for a particular component;

receiving a search response, wherein the search response presents the requested component according to its relative position in the hierarchical structure.

76. The method of claim 75, wherein the search response further presents each higher level component disposed between the requested component and a highest level component.

77. A system for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, comprising:

means for providing a visual presentation of the information model representing the software architecture; and

means for modifying the software architecture.

78. The system of claim 77, wherein the information model viewer further comprises:

means for presenting the data dependencies between components of the information model;

means for presenting the functional dependencies between components of the information model; and

means for presenting the control flow between program fragments contained within a component of the information model.

79. The system of claim 78, further comprising:

means for presenting a result of a search conducted within one or more information models or within one or more components of one or more information models.

80. The system of claim 79, further comprising:

means for presenting the information model as if the underlying body of source code was in a particular programming language.

81. The system of claim 77, wherein the system architect further comprises:

means for reorganizing the hierarchical component structure of the information model;

means for merging two or more components together into a compound component;

means for fragmenting a compound component into two or more components; and

means for adding new components to the information model.

82. The system of claim 81, wherein the system architect further comprises:

means for creating new components; and

means for relating the newly created components into a new information model having no underlying body of source code.

83. The system of claim 81, wherein the system architect further comprises:

means for extracting functionally related components of an information model; and

means for creating a new information model having a reduced set of components serving a desired function.

84. The system of claim 77, further comprising:

means for editing documentation associated with the information model; and

means for receiving new or modified documentation files associated with the information model.

85. The system of claim 77, further comprising:

means for editing source code files included with the body of source code; and

means for receiving new or modified source code files for inclusion with the body of source code.

86. The system of claim 85, wherein the means for receiving new or modified source code files further comprises:

means to provide source code files from the body of source code.

87. The system of claim 77, further comprising:

means for accepting a query;

means for searching the information model; and

means for providing results.

88. The system of claim 77, further comprising:

means for compiling a plurality of documentation objects into an information model document.

89. The system of claim 77, further comprising:

means for comparing at least two information models; and

means for determining the differences between the at least two information models.

90. A computer readable medium having stored thereon one or more sequences of instructions for causing one or more microprocessors to perform the steps for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, the steps comprising:

merging two or more components into a compound component.

91. The computer readable medium of claim 90 further comprising the step of:

modifying the hierarchical structure between two or more components in an information model.

92. A computer readable medium having stored thereon one or more sequences of instructions for causing one or more microprocessors to perform the steps for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, the steps comprising:

fragmenting a compound component into its constituent components.

93. The computer readable medium of claim 92 further comprising the step of:

modifying the hierarchical structure between two or more components in an information model.

94. A computer readable medium having stored thereon one or more sequences of instructions for causing one or more microprocessors to perform the steps for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, the steps comprising:

modifying the hierarchical structure between two or more components in an information model;

merging two or more components into a compound component; and

fragmenting a compound component into its constituent components.

95. The computer readable medium of claim 94, wherein the modifying step further comprises:

selecting a first component; and

promoting the first component in relation to a second component.

96. The computer readable medium of claim 94, wherein the modifying step further comprises:

selecting a first component; and

demoting the first component in relation to a second component.

97. The computer readable medium of claim 94, wherein the merging step further comprises merging a component and a first compound component into a second compound component.

98. The computer readable medium of claim 94, wherein the merging step further comprises merging a first compound component and a second compound component into a third compound component.

99. The computer readable medium of claim 94, wherein the fragmenting step further comprises fragmenting a compound component into two or more components.

100. The computer readable medium of claim 94, wherein the fragmenting step further comprises fragmenting a compound component into a component and a compound component.

101. The computer readable medium of claim 94, wherein the fragmenting step further comprises fragmenting a compound component into a first compound component and a second compound component.

102. The computer readable medium of claim 94, further comprising editing documentation associated with the information model through a documentation text editor.

103. The computer readable medium of claim 94, further comprising uploading new or modified documentation files through a file interface.

104. The computer readable medium of claim 94, further comprising compiling documentation associated with the information model into an information model document.

105. The computer readable medium of claim 94, further comprising:

creating a new component in the information model; and

organizing the new component within the hierarchical structure of the information model.

106. The computer readable medium of claim 94, further comprising searching the information model in response to a query and providing search results.

107. The computer readable medium of claim 94, further comprising:

comparing a first information model to a second information model; and

generating a difference set containing the differences between the first information model and the second information model identified by the comparison.

108. A system for manipulating an information model having a plurality of components arranged in a hierarchical structure representing a software architecture, comprising:

an information model viewer having a data dependency viewer capable of presenting the data dependencies between components of the information model and a functional dependency viewer capable of presenting the functional dependencies between components of the information model;

a system architect having an architect designer capable of reorganizing the hierarchical component structure of the information model;

an information model editor having a text interface capable of allowing editing of documentation associated with the information model and a file interface capable of receiving new or modified documentation files associated with the information model; and

an information model builder having a text interface capable of allowing editing of source code files included with the body of source code and a file

interface capable of receiving new or modified source code files for inclusion with the body of source code.

109. The system of claim 108, further comprising:

an information model viewer having a calling tree viewer capable of presenting the control flow between components of the information model and within a single component of the information model, and a language specific viewer capable of presenting the information model according to one or more programming language paradigms;

a system architect having an architect enhancer capable of adding new components to the information model, an architect creator capable of creating new components and relating the newly created components into a new information model having no underlying body of source code, an architect optimizer capable of extracting functionally related components of an information model and creating a new information model having a reduced set of components serving a desired function;

an information model search engine capable of accepting a query, searching the information model, and providing search results;

an information model document generator capable of compiling a plurality of documentation objects into an information model document; and

an information model difference generator capable of comparing at least two information models and determining differences between the at least two information models.